PROJEKTARBEIT
Modellierung, Simulation und Aufbau einer IoT-LED-Ansteuerung

Teil 1: Steuerung über Mikrofon

Autoren: Manh Linh Phan

Betreuer: Prof. Dr.-Ing. Jürgen Krumm

Technische Hochschule Georg Simon Ohm
Fakultät Elektrotechnik Feinwerktechnik Informationstechnik

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Das Klatschen von Händen ist einer der einfachsten perkussiven Klänge, da bei seiner Herstellung keine Werkzeuge oder Musikinstrumente verwendet werden. Es wird praktisch im alltäglichen menschlichen Leben verwendet, und diese Form des menschlichen Ausdrucks ist in fast jeder Kultur zu finden, insbesondere als arhythmisches Musikinstrument. Das Klatschen wird auch verwendet, um Übereinstimmung und Wertschätzung wiederzugeben, wenn es einige Sekunden lang rhythmisch von einer Gruppe wiederholt wird. Aus der Sicht eines Akustikers kann das Klatschen einzeln als Geräusch untersucht werden, das durch das Schlagen beider Hände zusammen oder gemeinsam als Klatschen einer Gruppe von Menschen erzeugt wird. Die Klangsynthese wird häufig in der Musik- und Filmindustrie sowie bei der Entwicklung von Spielen und Anwendungen für die virtuelle Realität eingesetzt. Beispielsweise können synthetische Klatschgeräusche von Hand Live-Aufzeichnungen in Sportspielen verbessern oder sogar ersetzen, indem sie die Modellierung des von einem großen Publikum erzeugten Applaus ermöglichen.

Clap-Switches finden Anwendungen in Bereichen wie Automatisierung und Sicherheit. Ein Klatschgeräusch kann als Auslösesignal entweder zum Ein- und Ausschalten der Relais oder als Warnsignal für einige Sicherheitssysteme verwendet werden. Aber die meisten von ihnen nutzen die Amplitude des Audiotons, um den Klatsch zu erkennen. Dies führt bei manchen Geräuschen zu dem gleichen Ergebnis wie beim Pfeifen oder Husten. Ziel dieses Teils der Projektarbeit ist, dass das Audiosignal offline (ohne Hilfe von der Verbindung mit anderen Geräten) zwischen Klatsch, Husten, Pfeifen, usw… klassifiziert wird.

2 Theorie

2.1 Der Algorithmus von der Arbeit

Das Eingangssignal ist das Pulse Code Modulation Audiosignal, das von dem PDM-Mikrofon aufgenommen wird und mit Hilfe von der PDM-PCM Schnittstelle von Cypress umgewandelt wird. Der Algorithmus wurde auf 2 Phasen verteilt. Im ersten Teil wird das Signal in kleine Frames untergeteilt. Danach wird das Signal in den Frequenzbereich umgewandelt unter der Verwendung von dem Fast Fourier Transform Algorithmus und mit der Mel Filterbank die passende Frequenzen herausgefiltert. Anschließend wird das Signal in ein zwei Dimension Array (Zeitbereich sowie Mel-/Frequenzbereich) umgewandelt. Danach wird der zweite Teil mit dem vorbereiteten Muster vergleichen. Das vorbereitete Muster wurden also durch den Transformationsprozess wie ein Echtzeitsignal behandelt. Der Dynamic Time Warping Algorithmus berechnet den Unterschied zwischen dem Echtzeitsignal, dem Muster und gibt den Abstand dazwischen zurück. Die k Muster mit dem kleinsten Abstand mit dem Echtzeitsignal wurde in eine Gruppe gekapselt. Die Mehrheit der Gruppe entscheiden das Ausgangssignal des ganzen Prozesses. Abbildung 2-1 zeigt das Flussdiagramm des implementierten Algorithmus.

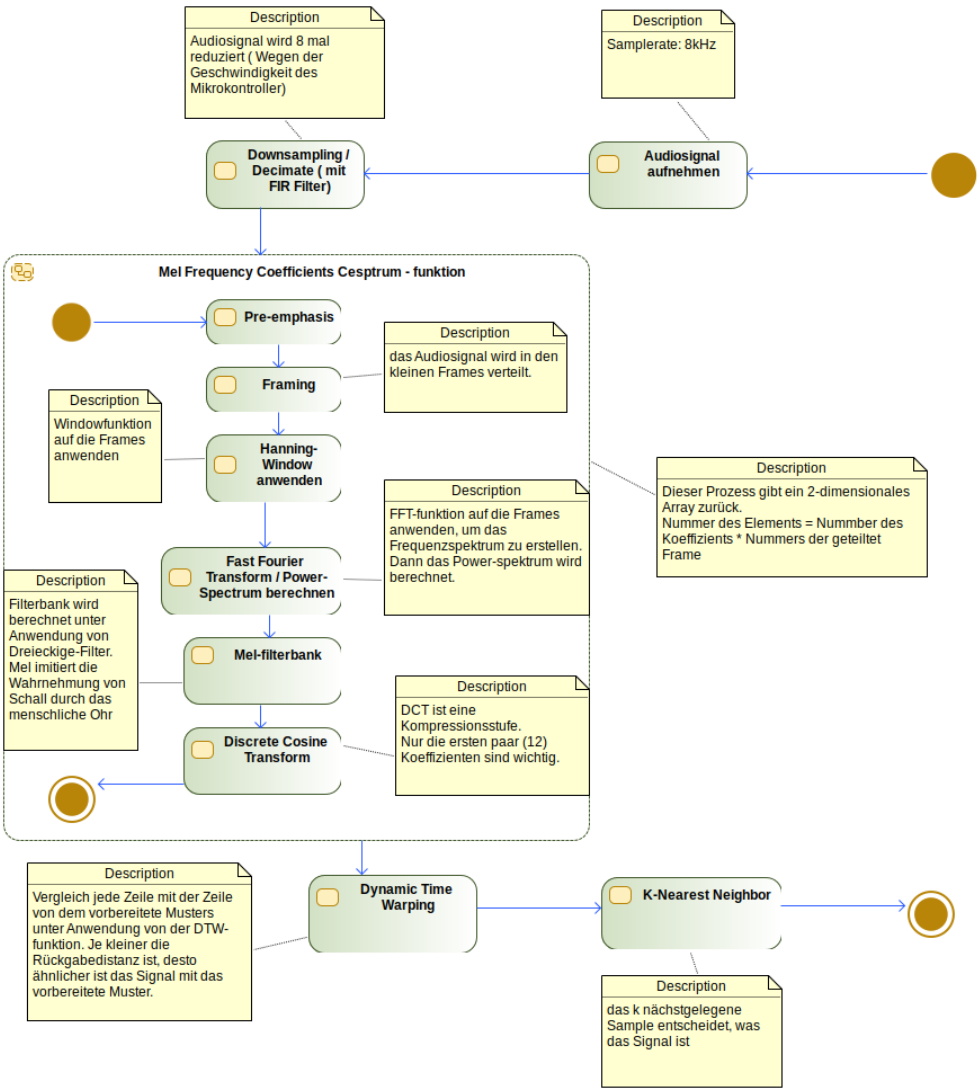


Abbildung 2-1: Flussdiagramm der Arbeit

2.2 Die Hardware-implemetierung

Das klatschaktivierte Schaltgerät kann grundsätzlich als niederfrequenter Schallimpuls-aktivierter Schalter beschrieben werden, der frei von Fehlauslösungen ist. Die Eingangskomponente ist ein Wandler, der das Eingang Klatschgeräusche empfängt und in elektrische Impulse umwandelt, die dann dem Psoc 62 zugeführt werden. Dieser Vorgang wird in Abbildung 2-2 gezeigt.

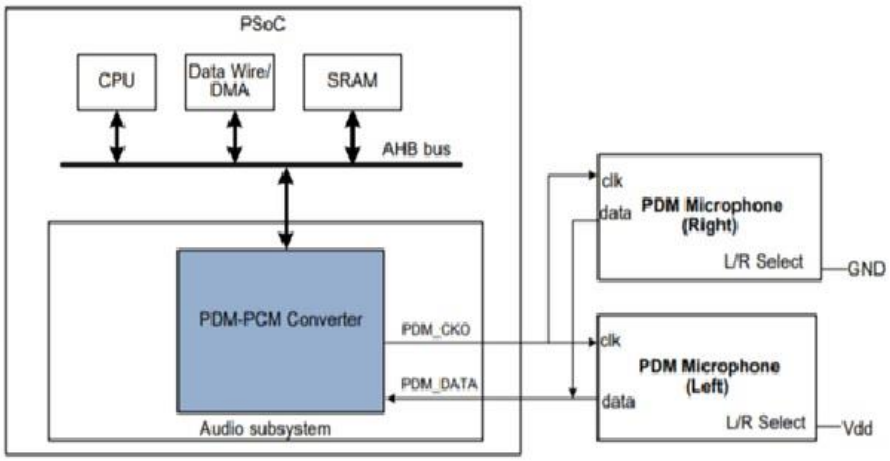


Abbildung 2-2: PDM-PCM-Schnittstelle [1]

2.2.1 Psoc 62

Die PSoC 62 MCU (CY8CPROTO-062-4343W) wurde für geringen Stromverbrauch und Leistung optimiert, wobei der Cortex-M0+ nur mit 15-µA/MHz und der Cortex-M4 nur mit 22 µA/MHz betrieben wird. Der Cortex-M0+ läuft bei 100MHz und der Cortex-M4 bei 150MHz. Außerdem unterstützt der Psoc 62 Audioschnittstellen wie PDM-PCM und I2S.

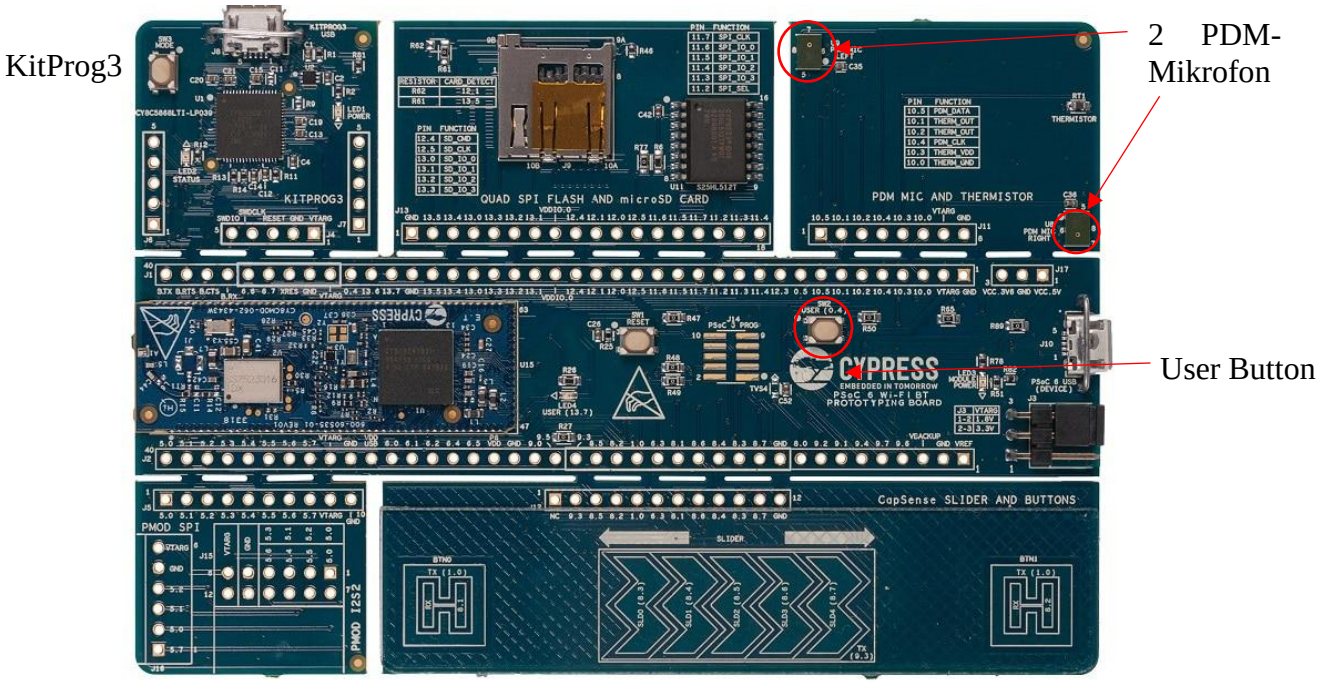


Abbildung 2-3: PSoC 6 Wi-Fi BT Prototyping Kit (CY8CPROTO-062-4343W) [2]

2.2.2 Modustoolbox DIE

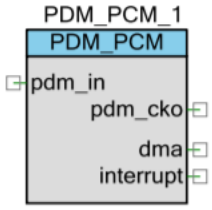
Die ModusToolbox ist ein Produkt der Firma Cypress Semiconductor. Die ModusToolbox IDE bietet Entwicklern Flexibilität beim Zugriff auf die branchenführenden Mikrocontroller (MCU) und drahtlosen Verbindungsgeräte von Cypress für das IoT. Diese IDE läuft über mehreren Host-Plattformen, einschließlich Windows, macOS und Linux.

2.2.3 PDM-Mikrofon und PDM/PCM Hardware Block

PDM ist die Abkürzung für Pulse Density Modulation. Das analoge Signal des Mikrofonelements wird zunächst verstärkt und dann mit hoher Rate 1-Bit- abgetastet und im PDM-Modulator quantisiert. Der Modulator kombiniert die Operationen der Quantisierung und des Rauschens. Formgebung der Ausgang ist ein Bitstrom, der von den einzelnen Bits bei der hohen Abtastrate kombiniert [3] würde.

PCM ist eine Methode zur Umwandlung eines analogen in ein digitales Signal. Informationen in analoger Form können von digitalen Computern nicht verarbeitet werden, so dass es notwendig ist, diese in digitale Form zu konvertieren.

Cypress bietet eine PDM_PCM_PDL-Komponente an, die einen Bitstrom von einer PDM-Quelle in PCM konvertiert, der ähnlich dem Ausgang eines ADCs ist. Die Eingang- and Ausgangsignale von der PDM/PCM Hardware wird in der unterliegenden Abbildung gezeigt.



Terminal Name	I/O Type	Description
pdm_in	Digital Input	PDM input signal from PDM device for conversion. Can be connected to digital input pin.
pdm_cko	Digital Output	Clock output signal for PDM sampling. Can be connected to digital output pin.
interrupt	Digital Output	Interrupt signal output. Visible when any interrupt source is activated.
dma*	Digital Output	DMA transfer request signal. Visible when the **DMA Trigger Enable** is selected.

Abbildung 2-4: Ein-/Ausgang der PDM-PCM Schnittstelle [1]

2.3 MFCC - Mel-Frequenz-Cepstrum-Koeffizienten

Der erste Schritt in jedem automatischen Spracherkennungssystem besteht darin, Merkmale zu extrahieren d.h. die Komponenten des Audiosignals zu identifizieren, die gut für die Identifizierung des sprachlichen Inhalts sind, und all die anderen Dinge zu verwerfen, die Informationen wie Hintergrundgeräusche, Emotionen usw. enthalten.

MFCCs sind die Basis für die automatische Sprach- und Sprechererkennung. Sie wurden von Davis und Mermelstein in den 1980er Jahren vorgestellt und gelten bis heute als Stand der Technik [4].

Framing

Das Signal wurde in Kurzzeit-Frames aufgeteilt. Der Grund für diesen Schritt ist, dass sich die Frequenzen in einem Signal mit der Zeit ändern, wenn man die Fourier-Transformation über das gesamte Signal anwendet, würden man die Frequenzkonturen des Signals im Lauf der Zeit verlieren. Um das zu vermeiden, kann man mit Sicherheit davon ausgehen, dass die Frequenzen in einem Signal über eine sehr kurze Zeitspanne stationär sind. Eine Fourier-Transformation wird über diesen kurzen Zeitraum durchführen, können wir eine gute Annäherung an die Frequenzkonturen des Signals durch die Verkettung benachbarter Rahmen erhalten.

Typische Frame-Größen in der Sprachverarbeitung reichen von 20 ms bis 40 ms mit 50% (+/-10%) Überlappung zwischen aufeinander folgenden Frames. Beliebte Einstellungen sind 25 ms für die Rahmengröße.

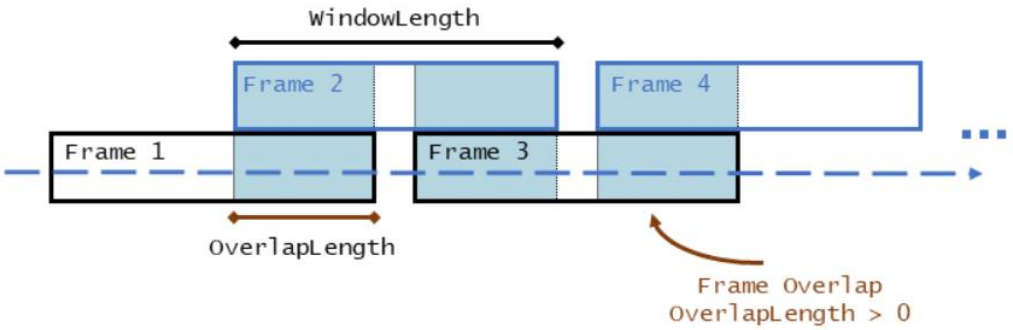


Abbildung 2-5: Überlappende Fenster [5]

2.3.1 Window

Die Fensterung wird im wesentlichen eingesetzt, um der Annahme der Fast-Fourier-Transformation, dass die Daten unendlich sind, deutlich entgegenzuwirken und um den spektralen Leckstrom zu reduzieren.

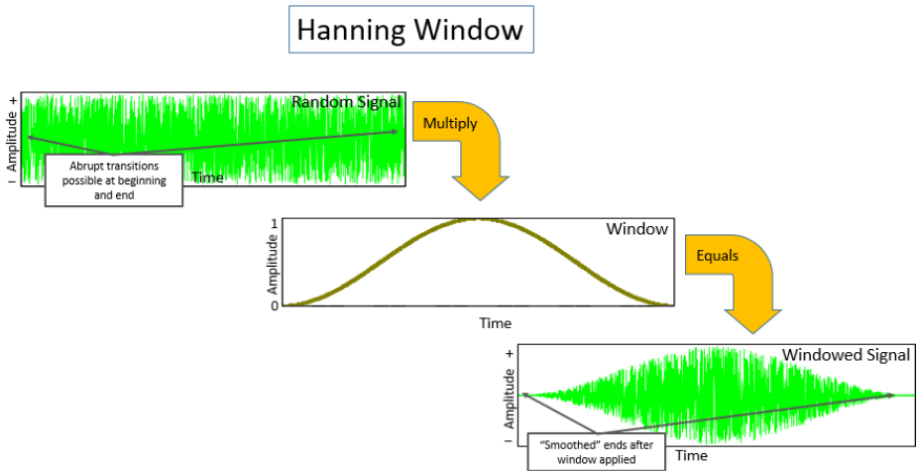


Abbildung 2-6: Hanning Window [6]

2.3.2 Fast Fourier Transform

Die Fourier-Transformation ist eine mathematische Technik, die eine Funktion der Zeit in eine Funktion der Frequenz transformiert.

Die schnelle Fourier-Transformation (FFT) ist ein diskreter Fourier-Transformationsalgorithmus, der die Anzahl der für N Punkte erforderlichen Berechnungen von $2 \cdot N^2$ auf $2 \cdot N \cdot \lg(N)$ reduziert, wobei $\lg$ der Logarithmus zur Basis 2 ist. Die Anzahl der Abtastpunkte ist die Potenz von 2. Die Algorithmen der schnellen Fourier-Transformation lassen sich im Allgemeinen in zwei Klassen einteilen: Dezimierung in der Zeit und Dezimierung in der Frequenz. Der Cooley-Tukey-FFT-Algorithmus ordnet zunächst die Eingangselemente in bit-umgekehrter Reihenfolge neu an und bildet dann die Ausgangstransformation (Dezimierung in der Zeit).

Sample numbers in normal order		Sample numbers after bit reversal	
Decimal	*Binary*	*Decimal*	*Binary*
0	0000	0	0000
1	0001	8	1000
2	0010	4	0100
3	0011	12	1100
4	0100	2	0010
5	0101	10	1010
6	0110	6	0100
7	0111	14	1110
8	1000	1	0001
9	1001	9	1001
10	1010	5	0101
11	1011	13	1101
12	1100	3	0011
13	1101	11	1011
14	1110	7	0111
15	1111	15	1111

FIGURE 12-3
The FFT bit reversal sorting. The FFT time domain decomposition can be implemented by sorting the samples according to bit reversed order.

Abbildung 2-7: Bit-Umkehrung [7]

Dieses einfache Flussdiagramm wird aufgrund seines geflügelten Aussehens als Schmetterling bezeichnet. Der Schmetterling ist das grundlegende Berechnungselement der FFT, das zwei komplexe Punkte in zwei weitere komplexe Punkte transformiert

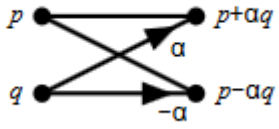


Abbildung 2-8: Die einfachste Butterfly-Formel [8]

Diagramm einer 8-Punkt-FFT mit $W = W_N = e^{-j\frac{2\pi}{N}}$.

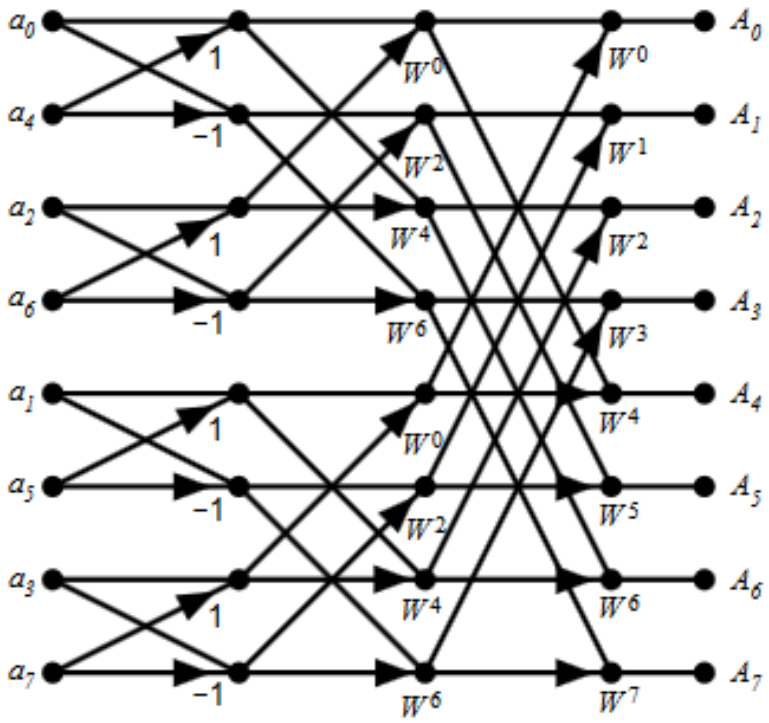


Abbildung 2-9: Diagramm einer 8-Punkt-FFT [8]

2.3.3 Mel filter bank

Mel-Skala: Die Mel-Skala ist eine Skala von Tonhöhen, die von den Hörern als gleich weit voneinander entfernt beurteilt werden. Der Bezugspunkt zwischen dieser Skala und der normalen Frequenzmessung wird durch die Gleichsetzung eines 1000-Hz-Tons, 40 dB über der Hörerschwelle, mit einer Tonhöhe von 1000 mels definiert. Unterhalb von etwa 500 Hz fallen die Mel- und Hertz-Skala zusammen; darüber werden immer größere Intervalle von den Zuhörern als gleichmäßige Tonhöhenschritte beurteilt.

Formel zur Umrechnung von f Hertz in m Mels:

$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right) = 1127 \ln\left(1 + \frac{f}{700}\right)$$

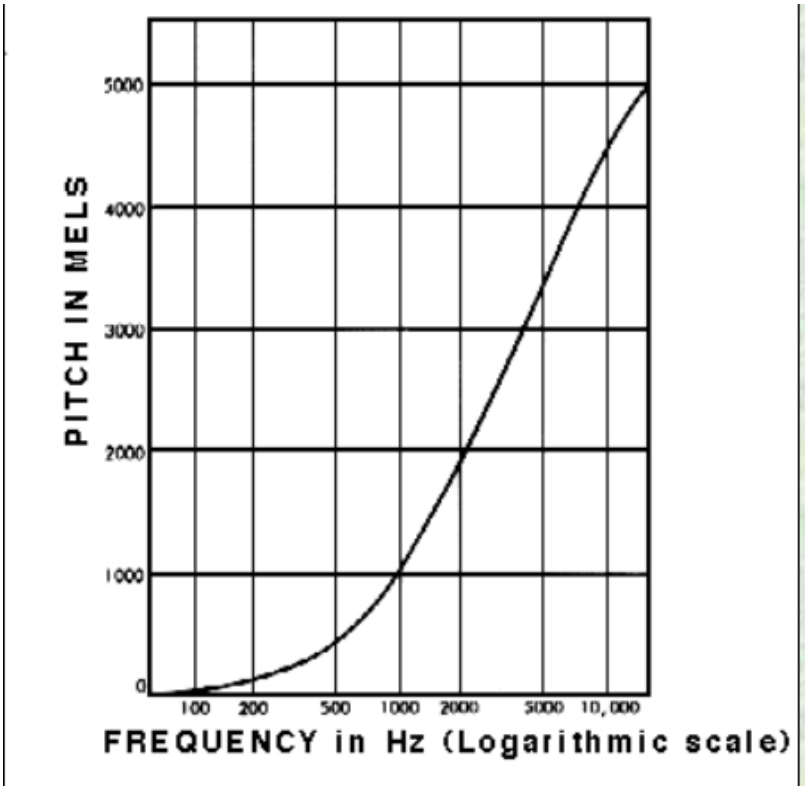


Abbildung 2-10: Mel Filter [9]

Jeder Filter in der Filterbank ist dreieckig mit einem Frequenzgang von 1 bei der Mittenfrequenz und nimmt linear gegen 0 ab, bis es die mittleren Frequenzen der beiden benachbarten Filter erreicht, wo der Frequenzgang 0 ist, wie in dieser Abbildung gezeigt:

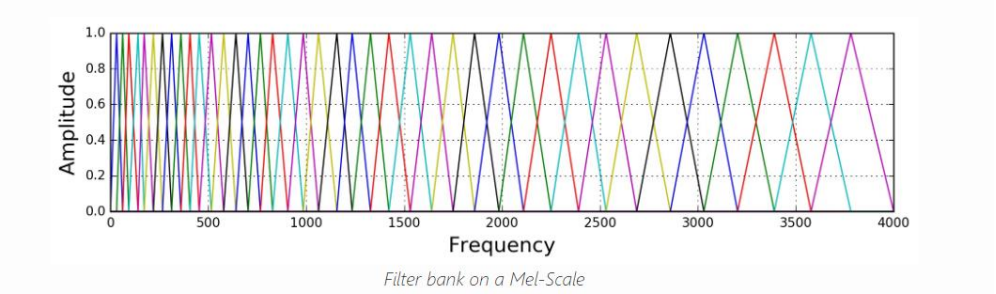


Abbildung 2-11: Filter Bank auf Mel-Skala [10]

2.3.4 Discrete Cosine Transform (DCT)

Es stellt sich heraus, dass die im vorherigen Schritt berechneten Filterbank-Koeffizienten hoch korreliert sind, was bei einigen Algorithmen des maschinellen Lernens problematisch sein könnte. Daher können wir die Diskrete Cosinus-Transformation (DCT) anwenden, um die Filterbank-Koeffizienten zu dekorrelieren und eine komprimierte Darstellung der Filterbänke zu erhalten. Typischerweise werden bei der automatischen Spracherkennung (ASR) die resultierenden Cepstralkoeffizienten 2-13 beibehalten und der Rest wird verworfen. Der Grund für das Verwerfen der anderen Koeffizienten ist, dass sie schnelle Änderungen der Filterbank-Koeffizienten darstellen und diese feinen Details nicht zur automatischen Spracherkennung (ASR) beitragen.

2.4 Mustervergleich

2.4.1 Dynamic Time Warping

Das Ziel von Zeitreihenvergleichsmethoden ist es, eine Abstandsmetrik zwischen zwei Eingabe-Zeitreihen zu erzeugen. Die Ähnlichkeit oder Unähnlichkeit von zwei Zeitreihen wird normalerweise durch die Umwandlung der Daten in Vektoren und die Berechnung des euklidischen Abstands zwischen diesen Punkten im Vektorraum berechnet. Die dynamische Zeitverzerrung ist eine bahnbrechende Zeitreihenvergleichstechnik, die seit den 1970er Jahren für die Sprach- und Worterkennung verwendet wird. [11]

Die dynamische Zeitverzerrung ermöglicht es, dass die beiden Kurven gleichmäßig übereinstimmen, obwohl die X-Achsen (d.h. die Zeit) nicht unbedingt synchron sind. Eine andere Möglichkeit ist, sich dies als einen robusten Unähnlichkeitsscore vorzustellen, bei dem eine niedrigere Zahl bedeutet, dass die Serie ähnlicher ist.

2.4.2 K-nearest neighbors algorithm (k-NN)

K nächste Nachbarn ist ein einfacher Algorithmus, der alle verfügbaren Fälle speichert und neue Fälle auf der Grundlage eines Ähnlichkeitsmaßes (z.B. Distanzfunktionen) klassifiziert. KNN wurde bereits Anfang der 1970er Jahre als nichtparametrisches Verfahren in der statistischen Schätzung und Mustererkennung eingesetzt.

Ein Fall wird durch ein Mehrheitsvotum seiner Nachbarn klassifiziert, wobei der Fall der Klasse zugeordnet wird, die unter seinen K nächsten Nachbarn, gemessen durch eine Entfernungsfunktion, am häufigsten vorkommt. Ist K = 1, dann wird der Fall einfach der Klasse seines nächsten Nachbarn zugeordnet.

Beispiel für K = 3. In diesem Fall gibt es 2 Dreiecke und 1 Quadrat in den 3 nächstgelegenen Figuren. Das heißt, der Muster ein Dreieck ist. Für K = 5 entscheidet sich der Muster als Quadrat

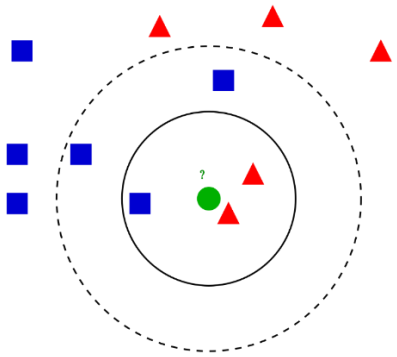


Abbildung 2-12: Figurbeispiel für k-nearest-neighbor Algorithmus [12]

3 Die Arbeitsimplementierung

3.1 Python und Audacity

Audacity und Python sind sowohl kostenlos als auch opensource. Ansonsten funktionieren sie in vielen Betriebssystemen (OS) wie Linux, Windows, MAC und haben eine Unterstützung durch die große Gemeinschaft, die kontinuierlich an ihrer Verbesserung arbeitet.

Mit Audacity lassen sich auf einfache Weise eine Reihe von Audiobearbeitungen und -aufnahmen durchführen. Der Benutzer kann die Samplerate oder die Bittiefe des Audiosignals wählen, ansonsten kann Audacity Wave-Dateien exportieren, die später in Python verwendet werden können.

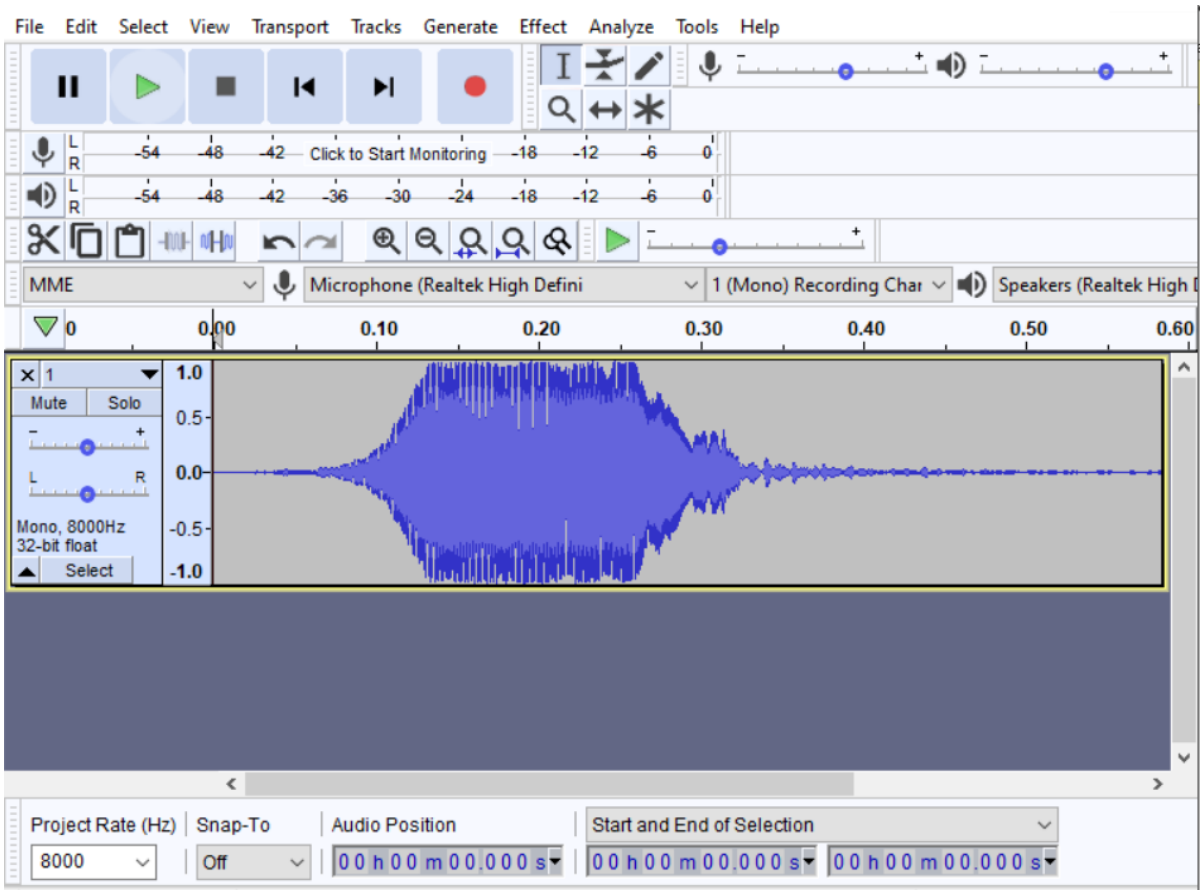


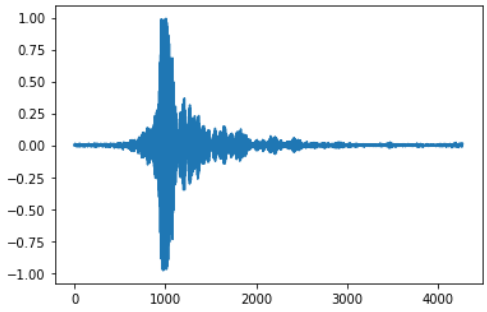
Abbildung 3-1: Audacity Interface

Python verfügt über eine große und robuste Bibliothek, sodass Python gegenüber anderen Programmiersprachen punktet, um den Algorithmus zu testen. Zum Beispiel kann die Funktion k-Nearest-Neighbor leicht vom Dictionary aus Python erstellt werden. Darüber hinaus kann Python problemlos neue Bibliotheken wie numpy, LibROSA [13] und matplotlib installieren. In diesem Projekt wurde die LibROSA zur Berechnung der schnellen Fourier-Transformation, der dynamischen Zeitverzerrung und des Mel-Frequenz-Cepstrums verwendet.

.

```
In [10]: n = 58
         WaveFile, sr = librosa.load(fpaths_src[n], sr = 8000)
         print(src_label[fpaths_src[n]])
         plt.plot(WaveFile)
         plt.show()
         MFCC = buildMfcc(WaveFile)
         librosa.display.specshow(MFCC,x_axis = 'time')
         plt.colorbar()

         Whistle
```




```
Out[10]: <matplotlib.colorbar.Colorbar at 0x26264d08be0>
```

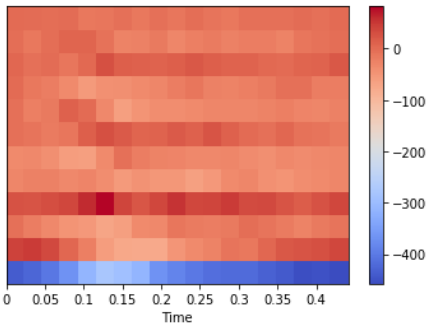


Abbildung 3-2: Beispiel für die Nutzung von LibROSA sowie Matplotlib in Jupyter Notebook

3.2 Gerätekonfigurator (Modus Toolbox)

Der Gerätekonfigurator ist Teil einer Sammlung von den Werkzeugen, die in der Modus Toolbox-Software enthalten sind. Um Geräteperipheriegeräte wie Takt und Pins sowie Standard-MCU-Peripheriegeräte zu aktivieren und zu konfigurieren, kann man mit Gerätekonfigurator verwenden. Ein eigenes Tool ist nicht erforderlich. Nach dem Konfigurieren und Speichern der Einstellungen eines bestimmten Geräts generiert der Gerätekonfigurator Firmware zur Verwendung in Ihrer Anwendung (Codegenerierung).

The PDM_PCM Component is configured for a sampling rate of 8 ksps. Configure the HFClk1 clock to a specific frequency (see below). The PDM_CLK frequency is calculated as:

$$\text{PDM CLK (kHz)} = \frac{\text{HFClk1 (kHz)}}{\text{1st Clock Divisor} \times \text{2nd Clock Divisor} \times (\text{3rd Clock Divisor} + 1)}$$

The sampling rate is calculated as:

$$\text{Sampling Rate (ksps)} = \frac{\text{PDM CLK}}{2 \times \text{Sinc Decimation Rate}}$$

$$\text{HFClk1} = \text{Sampling Rate} \times \text{1st Clock Divisor} \times \text{2nd Clock Divisor} \times (\text{3rd Clock Divisor} + 1) \times 2 \times \text{Sinc Decimation Rate}$$

Abbildung 3-3: Formel zur Berechnung der Taktfrequenz [14]

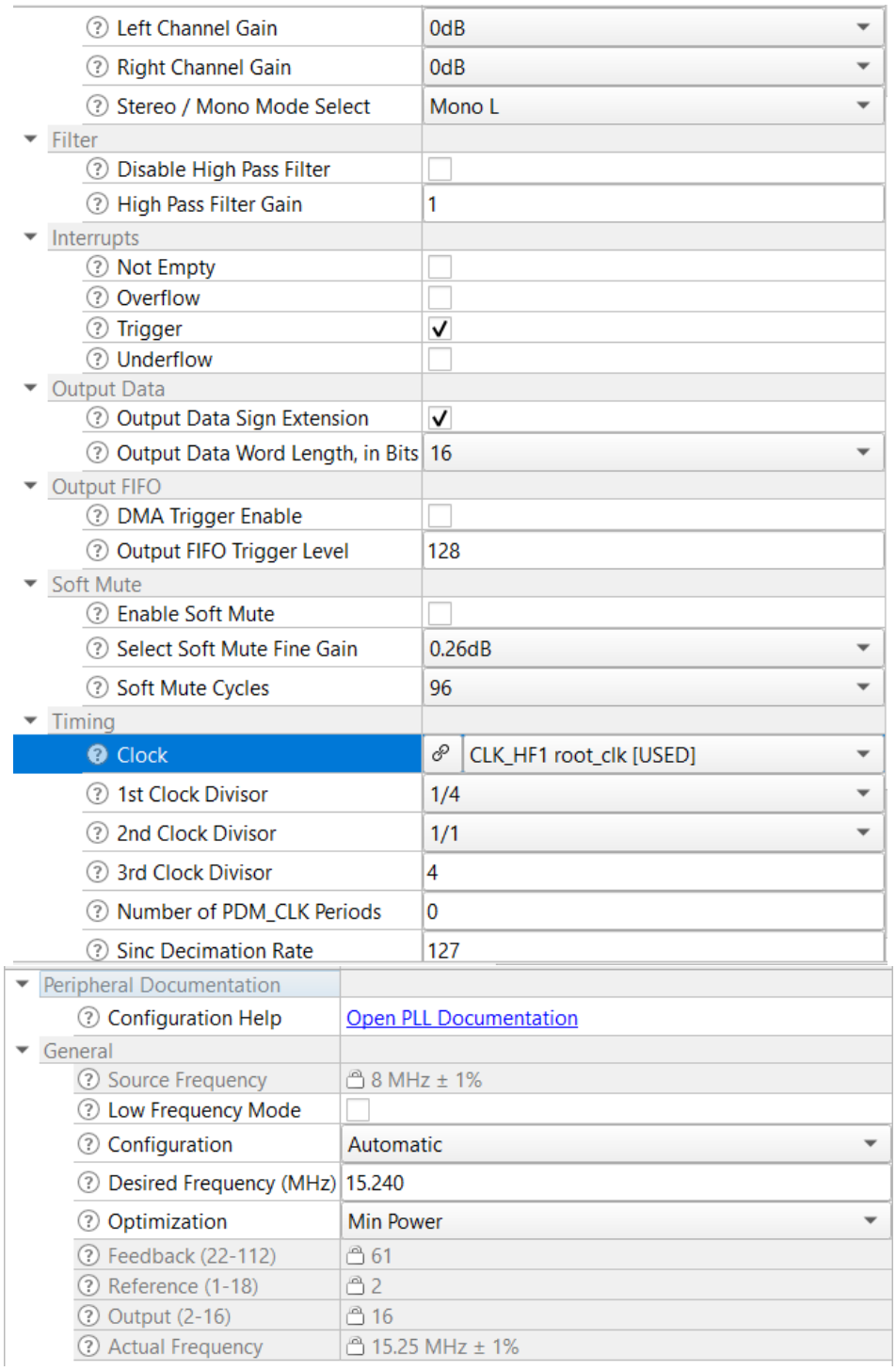


Abbildung 3-4: Beispiel Einstellung für Samplerate = 3ksps

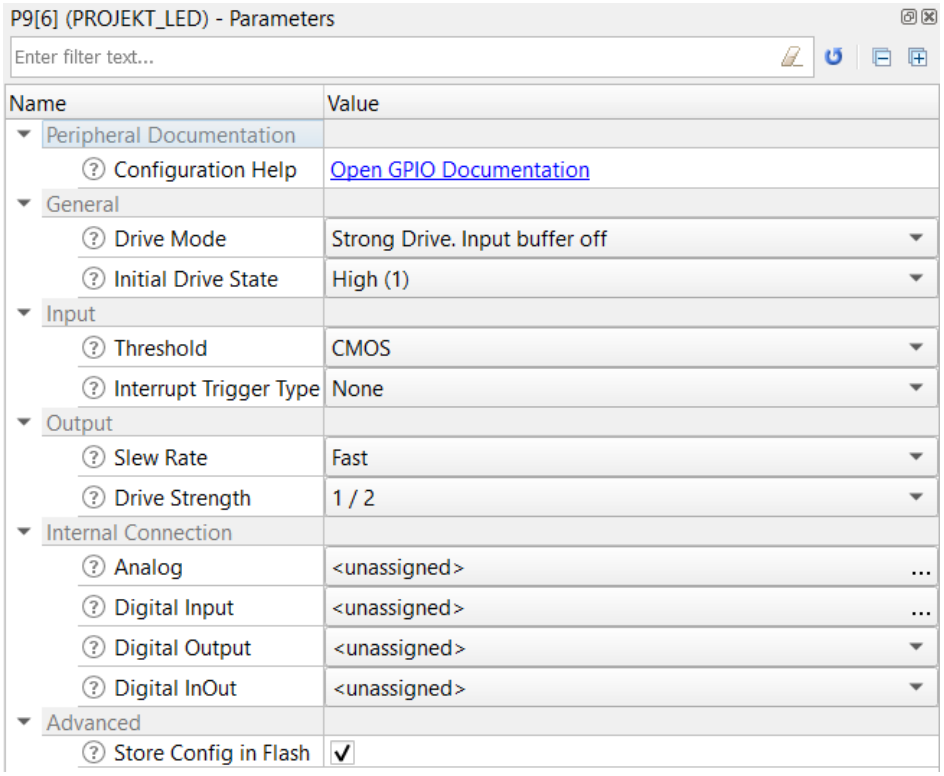


Abbildung 3-5: Einstellung für LED (Port 9 Pin 6)

Nach der Einstellung des Gerätekonfigurators, erstellt das Programm automatisch den Port sowie den Pin in der header-datei. Der Benutzer kann sofort mit den selbstdefinierten LED-Namen nutzen, ohne weiter Deklaration zu machen.

3.3 Interrupt

Aufgrund der Dauer des Transformationsprozesses ist der Prozess in 2 Fenster unterteilt. Das kleine Fenster ist für die Aufnahme des Audiosignals des Mikrofons. Das große Fenster ist der Arrayeingang für den Transformationsprozess. Wenn das kleine Fenster vollgeladen ist (die Füllzeit des kleinen Fensters muss größer sein als die Transformationszeit), wird der Transformationsprozess gestartet.

Der Ladevorgang aus dem kleinen Fenster wird in dem PDM-Mikrofon-Interrupt durchgeführt, so dass die Haupttransformation nicht beschädigt wird.

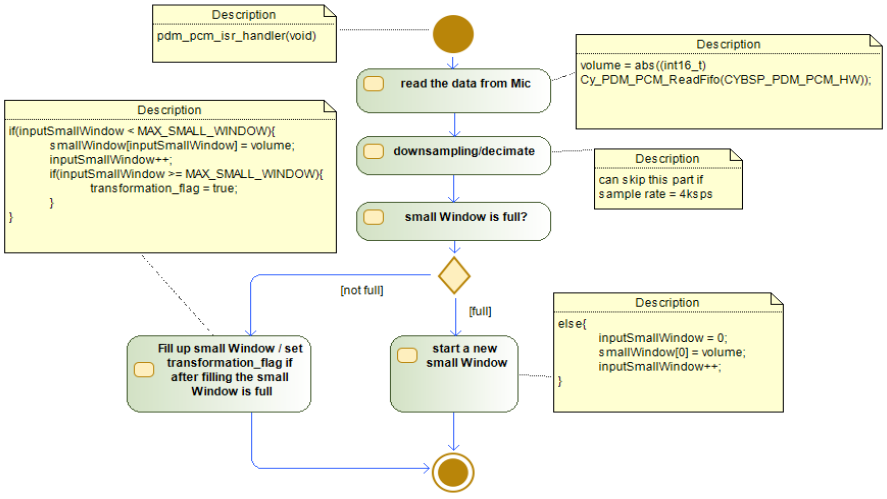


Abbildung 3-6: Interrupt Flow Diagramm

3.4 Mel-Frequency Cepstral Coefficients

3.4.1 Libmfcc [15] und KISSFFT [16]

Dies zwei Bibliotheken libmfcc und KISSFFT wurden zuerst gewählt. Der Vorteil von diesen Bibliotheken ist, dass beide in C geschrieben wurde und Open-Source sind. Außerdem gibt es viele Beispiele davon und sie sind einfach zu implementieren. Aber es stellte sich heraus, dass diese Bibliotheken auf Mikrokontrollern nicht gut funktionieren. Eine Transformationsprozess unter Verwendung von diesen Bibliotheken dauert etwa vier Sekunden, dass ist zu lange für die Echtzeitgeräuscherkennung.

3.4.2 CMSIS-DSP

CMSIS steht für Cortex Microcontroller Software Interface Standard. Sie stelle eine Bibliothek mit vielfältigen Funktionen zur Verfügung. Diese Bibliothek versorgt das effektive Einbinden einer Schnittstelle zwischen Anwendungssoftware und Mikrocontrollerhardware. Die Softwareentwicklung wird durch den Einsatz standardisierter Softwarefunktionen beschleunigt.

CMSIS besteht aus fünf ineinandergreifenden Spezifikationen, die die Codeentwicklung für alle Cortex-M-basierten Mikrocontroller unterstützen. Die fünf Spezifikationen sind wie folgt: CMSIS-Core, CMSIS RTOS, CMSIS DSP, CMSIS SVD und CMSIS DAP.

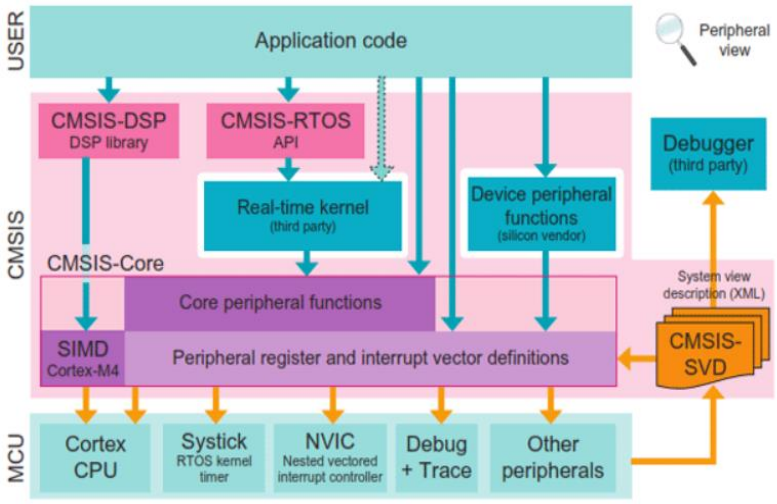


Abbildung 3-7: CMSIS Spezifikation [17]

		FFTs Per Second (More Is Faster)						
Inputs		Generic C					CMSIS	
N	Data	Arduino Uno	Arduino M0	Maple	Teensy 3.2	FRDM-K66F	Teensy 3.2	FRDM-K66F
32	Int16	201	3,127	8,052	9,671	14,493	30,864	58,824
64	Int16	99	1,594	4,230	4,892	7,353	16,529	33,333
128	Int16	*	639	1,644	1,922	2,865	7,156	14,286
256	Int16	*	325	854	970	1,458	3,669	7,353
512	Int16	*	135	*	399	594	1,575	3,226

		FFTs Per Second (More Is Faster)						
Inputs		Generic C					CMSIS	
N	Data	Arduino Uno	Arduino M0	Maple	Teensy 3.2	FRDM-K66F	Teensy 3.2	FRDM-K66F
32	Int32	80	1,247	5,230	5,187	9,804	16,474	31,250
64	Int32	*	597	2,714	2,607	5,263	8,578	16,667
128	Int32	*	236	1,031	1,007	1,953	3,293	6,211
256	Int32	*	114	531	506	1,031	1,675	3,155
512	Int32	*	47	*	206	406	677	1,272

		FFTs Per Second (More Is Faster)						
Inputs		Generic C					CMSIS	
N	Data	Arduino Uno	Arduino M0	Maple	Teensy 3.2	FRDM-K66F	Teensy 3.2	FRDM-K66F
32	Float32	157	457	1,499	2,015	21,277	3,542	40,000
64	Float32	*	199	649	880	12,658	1,811	21,277
128	Float32	*	81	263	351	4,608	658	9,524
256	Float32	*	37	117	159	2,695	331	4,878
512	Float32	*	16	*	67	1,019	130	2,123

* Insufficient RAM

Abbildung 3-8: FFT-Vergleichstabelle [18]

Im Vergleich zu KISSFFT ist die CMSIS-DSP-Bibliothek bei Verwendung des Datentyps Float32 doppelt so schnell.

3.4.3 MFCC Transformation und mit Muster vergleichen

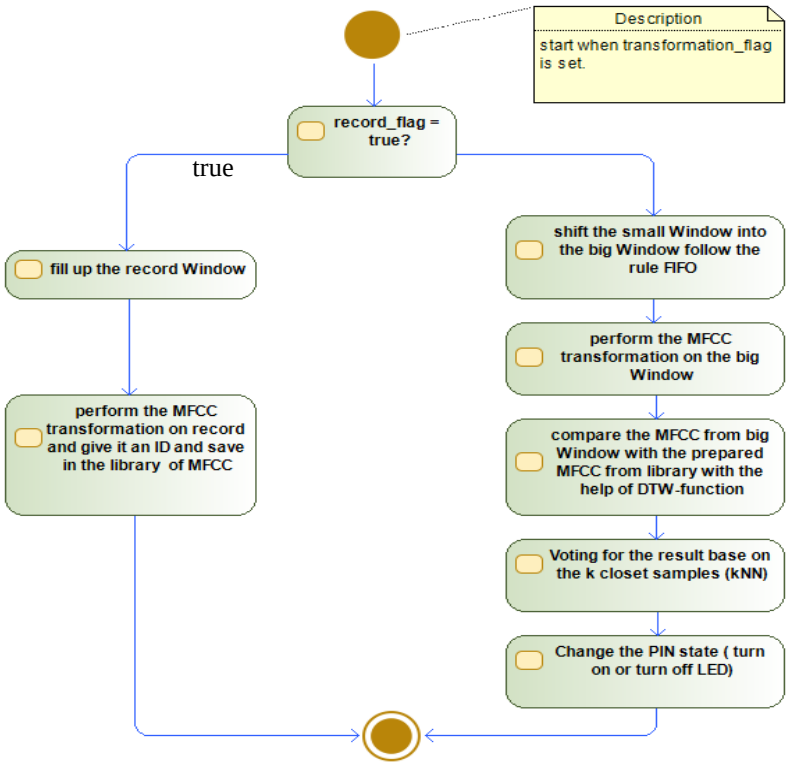


Abbildung 3-9: MFCC Transformation Flow Diagramm

Eine Struktur wird mit einigen Elementen erstellt:

- SoundID: um die Kennzeichnung des Tons zu speichern.
- Mfcc: 2-dimensionales Array zum Speichern des Koeffizienten.
- Dtw-distance: berechneter Abstand der Dynamic Time Warping Funktion.

Um die Transformation zu realisieren, wurden bei der Initialisierung eine Hanning Fenster Arrays, ein Array für die Mel Filterbank sowie eine Diskreter Cosinus Array erstellt. Bei der Änderung von einigen Makro Variables, kann man die Arraygröße des oben genannten Arrays sowie andere Eigenschaften wie Koeffizientenanzahl oder Grenzwert von Mel verändern.

Die Hauptfunktion nimmt das geteilt Frame als Eingangssignal und gibt die Mel Cepstral Koeffizienten zurück. Dort wurde elementweise Multiplikationen durchgeführt, um die Hanning Fenster anzuwenden. Das Signal wurde danach unter der Verwendung von Fast Fourier Transform Funktion der CMSIS-DSP von Zeitbereich nach Frequenzbereich transformiert. Nach Nyquist Theorie kann das Programm nur die Signale erkennen, die die Frequenz unter der Hälfte der Abtastrate liegt. Deswegen die obere halbe Frequenz werden verworfen und dann wird die Amplitude von der untere Frequenz berechnet. Nach Anwendung der Mel-Filterbank werden Signale von der gefiltert Frequenz in Mel gerechnet. Die Koeffizienten der Cepstral werden nach der Anforderung der Aufgabe ausgewählt (Hier werden die erste 12 Koeffiezienten gewählt).

Ein Array mit den k-Abständen wird nach dem k-nearest-neighbor Algorithmus erstellt, um das Audiosignal zu erkennen. Jede Schleife findet den größten berechneten Abstand und vergleicht ihn mit dem neu berechneten Abstand. Wenn der neue Abstand kleiner als der größte Abstand in der Matrix ist, ersetzen Sie ihn. Nachdem die Schleife beendet ist, zeigt die Funktion das Audiosignal an, das am meisten im Array auftritt.

Das Array von letzten N-Ausgangssignals verhindert ein einzelnes falsches Signal. Das heißt, dass nur wenn mehr als die Hälfte der Elementen im Array die gleiche gültige ID haben, gibt die Funktion die gültige ID zurück. Sonst gibt es die ID-Nummer 0 (ungültige ID) zurück. Der TON-Block verhindert eine kontinuierliche Änderung des LED-Status. Wenn das Programm ein gültiges Audiosignal erkennt, sperrt es die Veränderung für die nächste 1 Sekunde.

3.4.4 NOR Flash Memory

In diesem Projekt wird der Exemplar-Sound im SRAM (Static random-access memory) gespeichert, was bedeutet, dass die Daten nicht gespeichert werden, wenn die Karte nicht mit Strom versorgt wird.

Flash-Speicher auf der Basis von EEPROM - elektrisch löschbarer, programmierbarer Festwertspeicher basiert. Ein Flash-Speicher kann elektrisch gelöscht und neu programmiert werden. Der Unterschied zwischen Flash-Speicher und SRAM besteht darin, dass Flash zum Speichern von Daten keinen Strom benötigt.

In diesem Projekt bietet das Psoc 6 Wi-FI BT Prototyping Kit den 512-Mb Quad Serial Peripheral Interface NOR Flash-Speicher, der als der leistungsstärkste, sicherste und einfachste Speicher der Branche bekannt ist. Mit diesen Vorteilen ist der NOR-Flash-Speicher der ideale Speicher für die Codespeicherung in eingebetteten Systemen. Ansonsten kann die MicroSD-Kartenschnittstelle auch als externer Flash-Speicher verwendet werden, wenn der interne Flash-Speicher nicht ausreicht.

Cypress bietet einige Funktionen, damit kann der Benutzer einfach die Flash Memory anwenden. Das Array in RAM wurde von oder nach Flash in byteweise (uint8_t) kopiert. Hier muss der Benutzer die Adresse in Flash Memory definieren, sowie wie viele Bytes er in dem Flash Memory braucht.

4 Das Fazit

Das war ein herausfordernder, aber interessanter Teil der Projektarbeit. Am Anfang kann man viele Beispiels für die Klatsch-erkennung in dem Internet finden, aber die Informationen sind nicht ausreichend, weil es nur die Amplitude des Audiosignals benutzt. Die Lösung dafür kann man unter den Schlüsselwort Spracherkennung finden. Das nächste Problem liegt darin, die beste passende Bibliothek zu finden. Weil der Code im Mikrokontroller läuft, spielt die Geschwindigkeit des Codes eine große Rolle. Das letzte und schwierigste Problem ist wie man zwischen der Genauigkeit und Geschwindigkeit ausgleicht. Durch die Einstellung von 8000 Sample-Rate und entsprechendem Überlappungsfenster, sowie der Fast Fourier Transform Funktion, konnten einige der Audiosignale und auch einige kurze Wörter bereits erkannt werden. Aber die Geschwindigkeit des Mikrocontrollers begrenzt die Genauigkeit dieses Algorithmus. Nach den Tests wurden 400 0 Abtastraten und 20 Mal Erkennung pro Sekunde gewählt. Das bedeutet, dass der Mikrocontroller nur das Signal mit einer maximalen Frequenz von 2khz erkennen kann. Pfeifen wurde stattdessen Klatschen auswählt, weil es im passenden Frequenzbereich liegt, sowie lange und stabil ist.

In diesem Projekt, mit Hilfe von Dynamic Time Warping Algorithmus, kann man schon gut eine kleine Menge von Audiosignale erkennen. Der Nachteil des Algorithmus ist, dass man eine große Menge von Muster vorbereiten muss. Außerdem je mehr Muster man hat, desto öfters muss das Echtzeitsignal mit den Mustern verglichen werden. Für eine Erweiterung kann man neuronale Netzwerke oder das Hidden Markov Modell anwenden. Das Problem ist das, dass diese zwei Methoden eine große Anzahl von Audiosignalen braucht, weil sie davon die Eigenschaften sowie die versteckten Zustände prognostizieren. Aber sie bieten einen großen Vorteil, dass wir nicht nur das einzelne Geräusch sowie einzelne Wort erkennen können, sondern auch ganze Sätze.

5 Literaturangaben

[1] Cypress Semiconductor Corp.: Pulse-Density Modulation to Pulse-Code Modulation Decoder (PDM_PCM_PDL), 2020. https://www.cypress.com/documentation/component-datasheets/pulse-density-modulation-pulse-code-modulation-decoder-pdmpcmpdl, abgerufen am: 28.05.2020

[2] Cypress Semiconductor Corp.: PSoC 6 Wi-Fi BT Prototyping Kit (CY8CPROTO-062-4343W), 2020. https://www.cypress.com/documentation/development-kitsboards/psoc-6-wi-fi-bt-prototyping-kit-cy8cproto-062-4343w, abgerufen am: 28.05.2020

[3] Wikipedia: Pulse-density modulation, 2020. https://en.wikipedia.org/w/index.php?title=Pulse-density_modulation&oldid=954286453, abgerufen am: 29.05.2020

[4] Davis, S. u. Mermelstein, P.: Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. IEEE Transactions on Acoustics, Speech, and Signal Processing 28 (1980) 4, S. 357–366

[5] The MathWorks: Extract mfcc, log energy, delta, and delta-delta of audio signal - MATLAB mfcc, 2020. https://www.cypress.com/file/377781/download, abgerufen am: 28.05.2020

[6] Peter Schaldenbrand: Window Types: Hanning, Flattop, Uniform, Tukey, and Exponential, 2020. https://community.sw.siemens.com/s/article/window-types-hanning-flattop-uniform-tukey-and-exponential, abgerufen am: 28.05.2020

[7] Smith, S. W.: The scientist and engineer's guide to digital signal processing. San Diego, Calif.: California Technical Publ 1997

[8] Paul Heckbert: Fourier transforms and the fast Fourier transform (FFT) algorithm. 1995

[9] Appleton, J. H., Perera, R. C. u. Luening, O. (Hrsg.): The development and practice of electronic music. Englewood Cliffs, NJ: Prentice-Hall 1975

[10] Fayek, H.: Speech Processing for Machine Learning: Filter banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between, 2016. https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html, abgerufen am: 28.05.2020

[11] Portilla, R.: Understanding Dynamic Time Warping. Databricks (2019)

[12] K Nearest Neighbor | KNN Algorithm | KNN in Python & R, 2018. https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/, abgerufen am: 28.05.2020

[13] librosa development team: LibROSA — librosa 0.7.2 documentation, 2020. https://librosa.github.io/librosa/, abgerufen am: 28.05.2020

[14] Cypress Semiconductor Corp.: CE219431 - PSoC 6 MCU PDM-to-PCM Example, 2020. https://www.cypress.com/documentation/code-

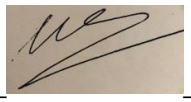
examples/ce219431-psoc-6-mcu-pdm-pcm-example, abgerufen am: 28.05.2020
[15] Jeremy Sawruk: jsawruk/libmfcc, 2020. https://github.com/jsawruk/libmfcc, abgerufen am: 28.05.2020
[16] Mark Borgerding: mborgerding/kissfft, 2020. https://github.com/mborgerding/kissfft, abgerufen am: 28.05.2020
[17] Embedded by AspenCore: Basics of the Cortex MCU Software Interface Standard: Part 1 - CMSIS Specification - Embedded.com, 2015. https://www.embedded.com/basics-of-the-cortex-mcu-software-interface-standard-part-1-cmsis-specification/, abgerufen am: 28.05.2020
[18] Chip: Open Audio: Benchmarking - FFT Speed, 2020. http://openaudio.blogspot.com/2016/09/benchmarking-fft-speed.html, abgerufen am: 28.05.2020

Eigenständigkeitserklärung:

Ich versichere, dass ich die vorgelegte Projektarbeit eigenständig und ohne fremde Hilfe verfasst, keine anderen als die angegebenen Quellen verwendet und die den benutzten Quellen entnommenen Passagen als solche kenntlich gemacht habe. Diese Projektarbeit ist in dieser oder einer ähnlichen Form in keinem anderen Kurs vorgelegt worden.

Name, Vorname: Phan, Manh Linh

Nürnberg, den 21.06.2020

Unterschrift: _____  _____